Improved Accuracy for MR3-based Eigensolvers

Matthias Petschow and **Paolo Bientinesi**

AICES, RWTH Aachen
pauldj@aices.rwth-aachen.de

SIAM Conference on Computational Science and Engineering
February 26th, 2013
Boston, MA

Hermitian dense eigenproblems

$AX = X\Lambda$	STDEIG
$AX = XB\Lambda$	GENEIG

- Input:

$A \in \mathcal{C}^{n\times n}$	$A^H = A$
$B \in \mathcal{C}^{n\times n}$	SPD
$k,\ 1 \le k \le n$	#eigenpairs

- Output:

$X \in \mathcal{C}^{n\times k}$	eigenvectors
$\Lambda \in \mathcal{R}^{k\times k}$	eigenvalues

- Accuracy:

$\|AX - X\Lambda\|$	residual
$\|X^H X - I\|$	**orthogonality**

Nested eigensolvers

GENEIG $\rightarrow$ STDEIG $\rightarrow$ TRDEIG

1	$LL^H = B$	Cholesky factorization	$O(n^3)$
2	$M \leftarrow L^{-1}AL^{-H}$	Reduction to standard form	$O(n^3)$
3	$T = Q^H M Q$	Reduction to tridiagonal form	$O(n^3)$
4	$TZ = Z\Lambda$	Tridiagonal eigenproblem	$O(kn)$–$O(n^3)$
5	$Y = QZ$	Backtransformation #1	$O(kn^2)$
6	$X = L^{-H}Y$	Backtransformation #2	$O(kn^2)$

Tridiagonal eigenproblem

Algorithms

1958	Bisection + Inverse Iteration (BI)	subsets	$O(kn^2)$
1961	QR	robust, accurate	$O(n^3)$
1981	Divide & Conquer (DC)	BLAS3, accurate	$O(n^3)$
1997	MR3 / MRRR	subsets, fast	$\mathbf{O(kn)}$

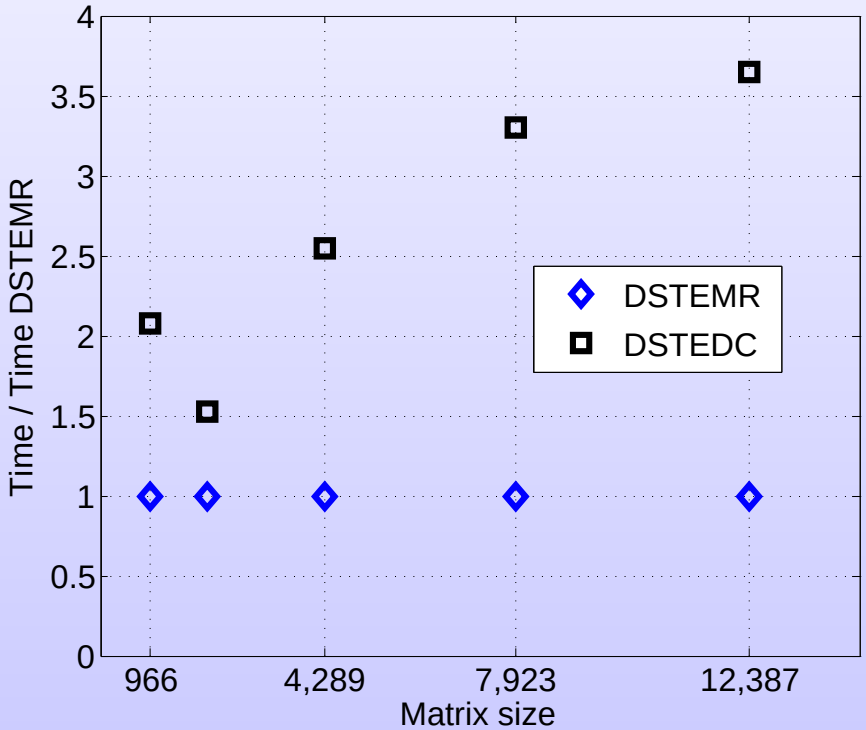
Tridiagonal eigenproblem

Algorithms

1958	Bisection + Inverse Iteration (BI)	subsets	$O(kn^2)$
1961	QR	robust, accurate	$O(n^3)$
1981	Divide & Conquer (DC)	BLAS3, accurate	$O(n^3)$
1997	**MR3 / MRRR**	subsets, fast	$\mathbf{O(kn)}$

Speed & accuracy

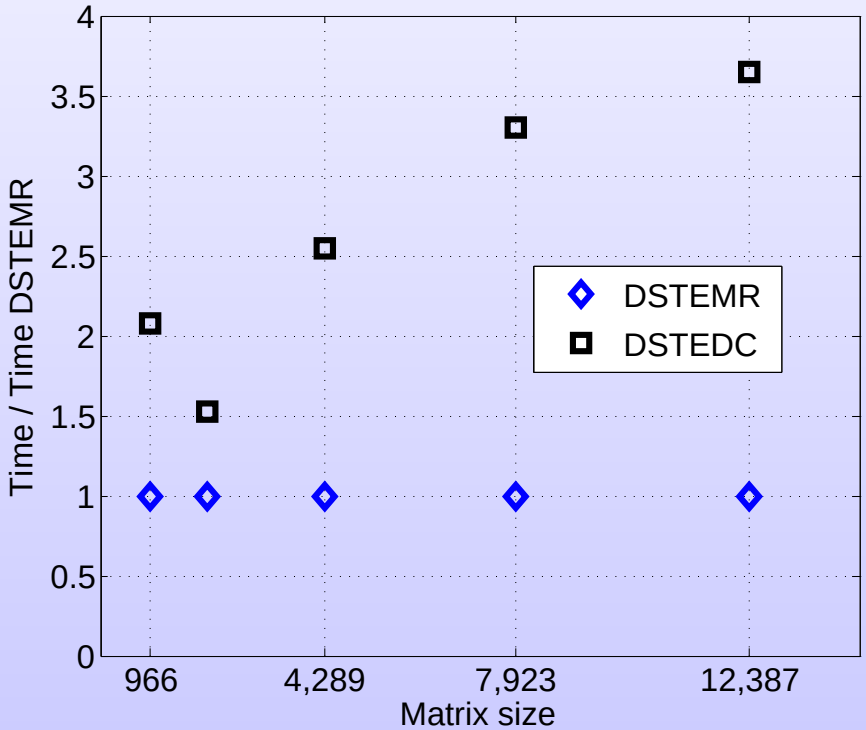
application matrices, full spectrum, double precision



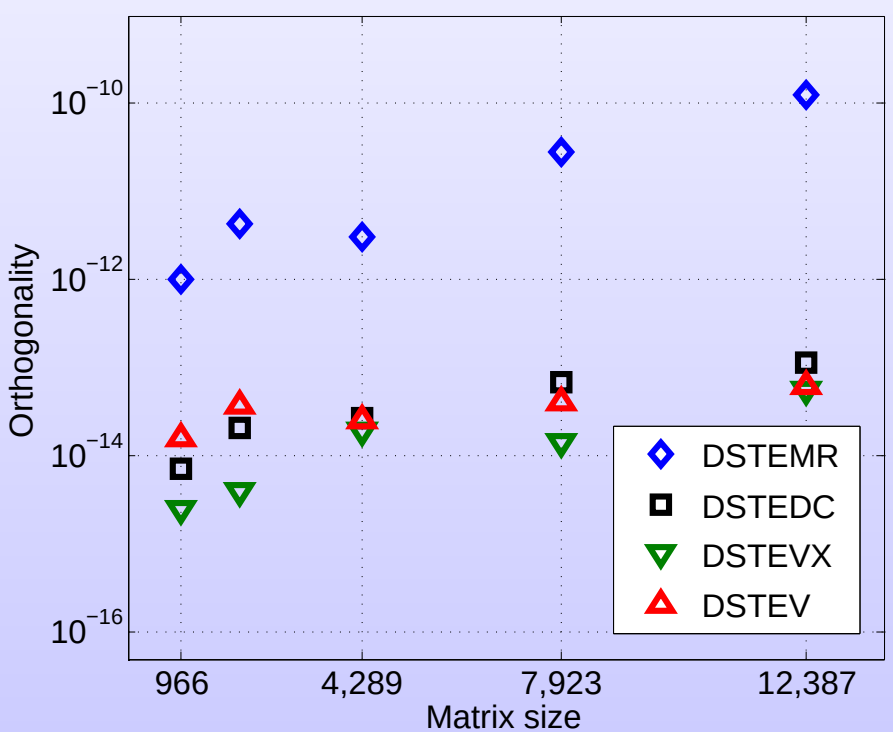
(a) DC vs. MR3

Speed & accuracy

application matrices, full spectrum, double precision



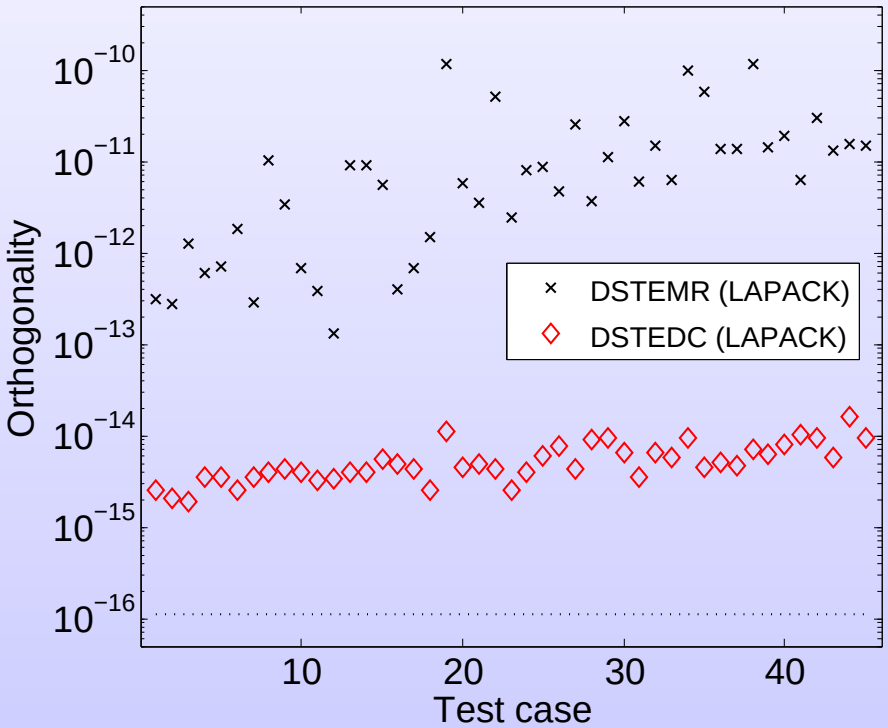
(a) DC vs. MR3



(b) $\max_{i \neq j} |z_i^H z_j|$

accuracy(tridiagonal) $\Rightarrow$ accuracy(dense)

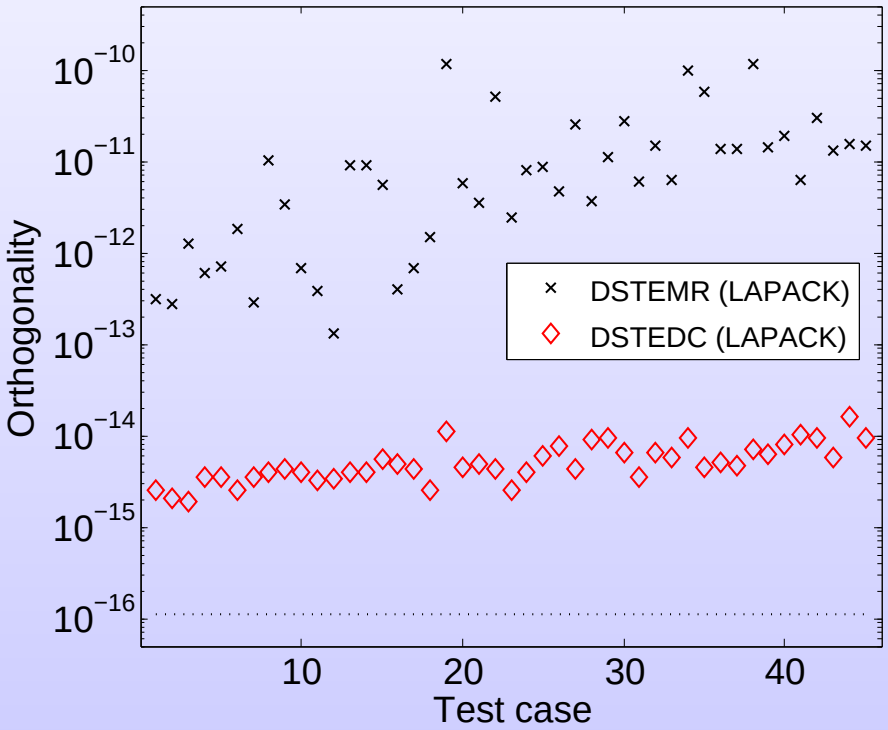
application matrices, size $\in [1.000, \ldots, 8.000]$



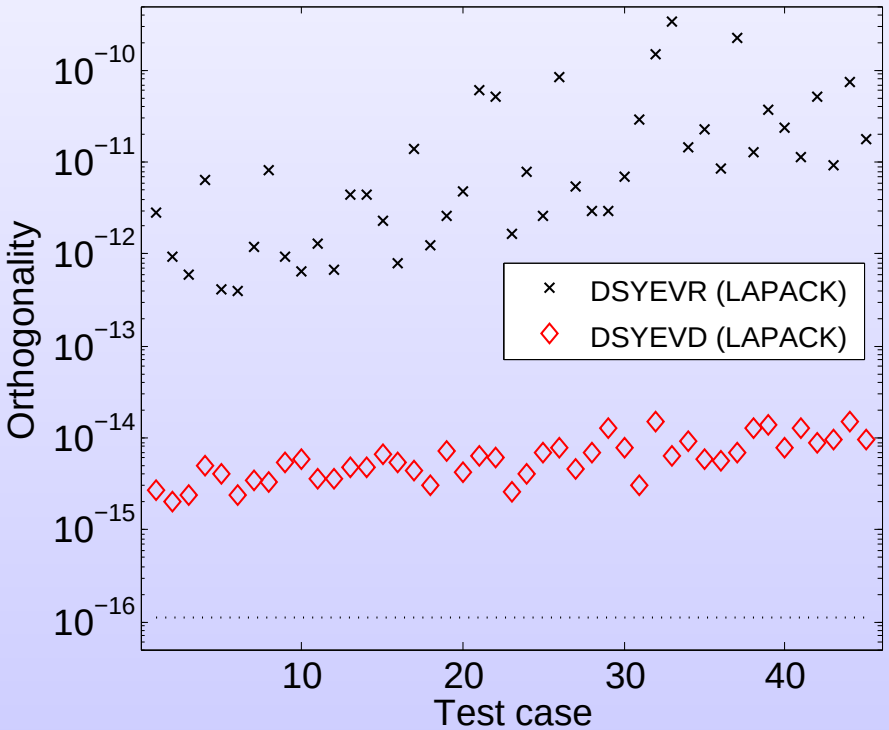
(a) Tridiagonal

accuracy(tridiagonal) $\Rightarrow$ accuracy(dense)

application matrices, size $\in [1.000, \ldots, 8.000]$



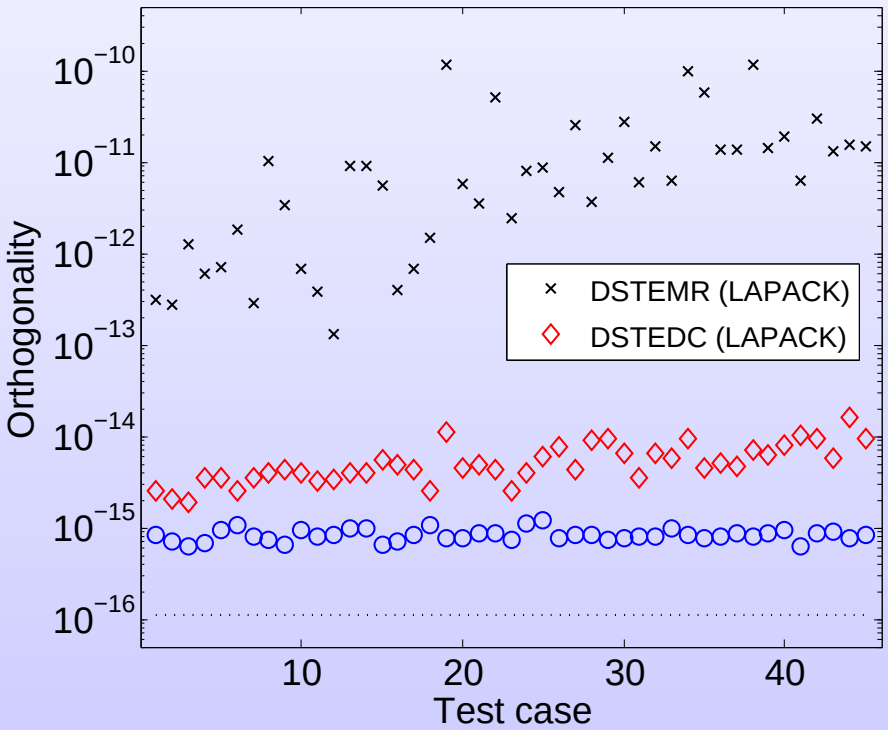
(a) Tridiagonal



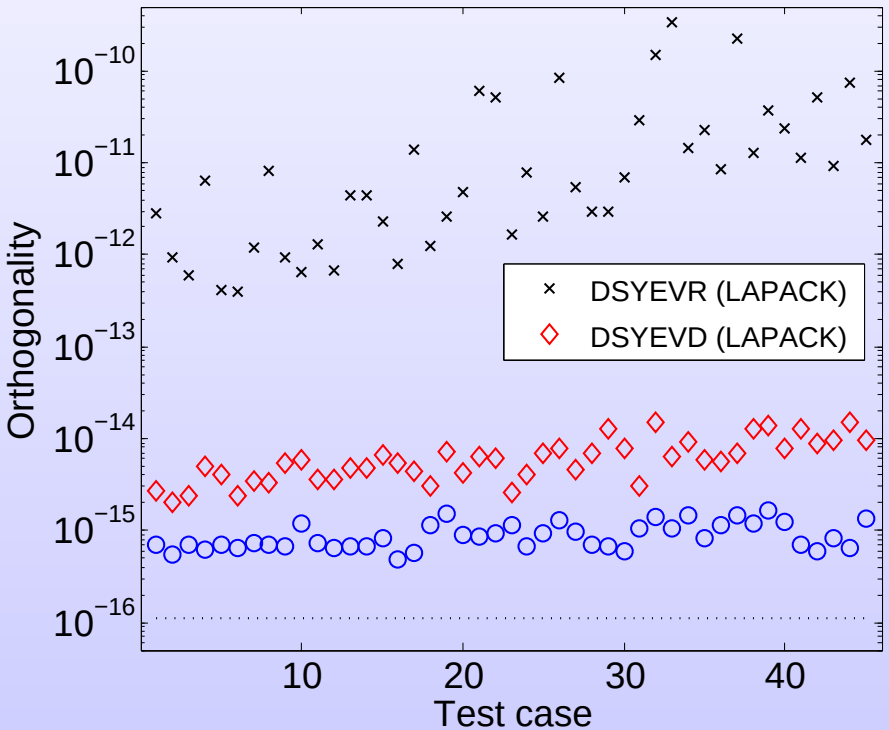
(b) Dense

where are we heading?

○ : improved MR3 — at what cost?



(a) Tridiagonal



(b) Dense

2 MR3 — trading speed for accuracy

MRRR – Dhillon & Parlett (1998)

Multiple Relatively Robust Representations

MRRR – Dhillon & Parlett (1998)

Multiple Relatively Robust Representations

- k eigenpairs in $O(nk)$ operations

MRRR – Dhillon & Parlett (1998)

Multiple Relatively Robust Representations

- k eigenpairs in $O(nk)$ operations
- no reorthogonalization

MRRR – Dhillon & Parlett (1998)

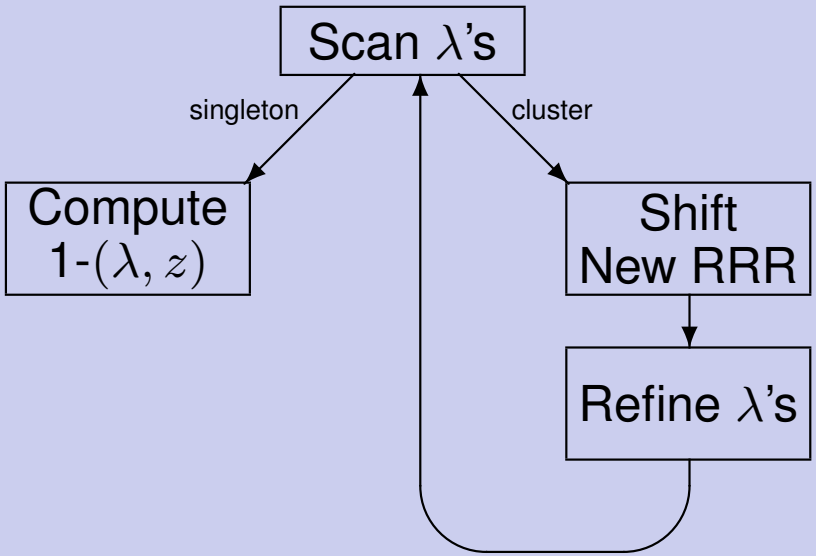
Multiple Relatively Robust Representations

- k eigenpairs in $O(nk)$ operations
- no reorthogonalization
- 1) eigenvalues (*dqds*, *bisection*)
  2) eigenvectors + eigenvalues

MRRR – Dhillon & Parlett (1998)

Multiple Relatively Robust Representations

- k eigenpairs in $O(nk)$ operations
- no reorthogonalization
- 1) eigenvalues (*dqds*, *bisection*)
  2) eigenvectors + eigenvalues



Low accuracy. How to cure MR3?

Low accuracy. How to cure MR3?

- Adjusting internal parameters and thresholds
  - random perturbations
  - thresholds (element growth)
  - eigenvalue refinement
  - initial eigenvalue approximation
  - minimum relative gap: `gaptol`
  - stopping criteria (BX, RQI)

Low accuracy. How to cure MR3?

- Adjusting internal parameters and thresholds
  - random perturbations
  - thresholds (element growth)
  - eigenvalue refinement
  - initial eigenvalue approximation
  - minimum relative gap: `gaptol`
  - stopping criteria (BX, RQI)

- Data conversion

  single precision $\rightarrow$ double precision

  double precision $\rightarrow$ extended precision

  double precision $\rightarrow$ quad precision

Low accuracy. How to cure MR3?

- Adjusting internal parameters and thresholds
  - random perturbations
  - thresholds (element growth)
  - eigenvalue refinement
  - initial eigenvalue approximation
  - minimum relative gap: `gaptol`
  - stopping criteria (BX, RQI)

- Data conversion

  single precision $\rightarrow$ double precision
  double precision $\rightarrow$ extended precision
  double precision $\rightarrow$ quad precision

No: both memory & performance issues

Low accuracy. How to cure MR3?

- Adjusting internal parameters and thresholds
  - random perturbations
  - thresholds (element growth)
  - eigenvalue refinement
  - initial eigenvalue approximation
  - minimum relative gap: `gaptol`
  - stopping criteria (BX, RQI)

- Data conversion

  single precision $\rightarrow$ double precision
  double precision $\rightarrow$ extended precision
  double precision $\rightarrow$ quad precision

  No: both memory & performance issues

- Mixed precision

  input/output precision: ϵ_x
  internal precision: ϵ_x and ϵ_y, with $\epsilon_y < \epsilon_x$

Internal parameters

Internal parameters

gaptol

$$\texttt{if} \min_{j \neq i} \frac{|\lambda_i - \lambda_j|}{|\lambda_i|} \geq \texttt{gaptol} \quad \texttt{then} \quad \lambda_i \text{ is a singleton}$$

Small	$\Rightarrow$	less clustering, better robustness, more parallelism, more BX
Large	$\Rightarrow$	more work, deeper trees, better orthogonality (?), more failures

Internal parameters

gaptol

$$\texttt{if} \min_{j \neq i} \frac{|\lambda_i - \lambda_j|}{|\lambda_i|} \geq \texttt{gaptol} \quad \texttt{then} \quad \lambda_i \text{ is a singleton}$$

Small $\Rightarrow$ less clustering, better robustness, more parallelism, more BX

Large $\Rightarrow$ more work, deeper trees, better orthogonality (?), more failures

$$|\hat{z}_i^H \hat{z}_j| \quad \leq \quad C_1 \left(k_{rs} + k_{rr}\, d_{max}\right) \frac{n\epsilon_d}{gaptol} \quad \leq \quad C_2 \frac{n\epsilon_d}{gaptol}$$

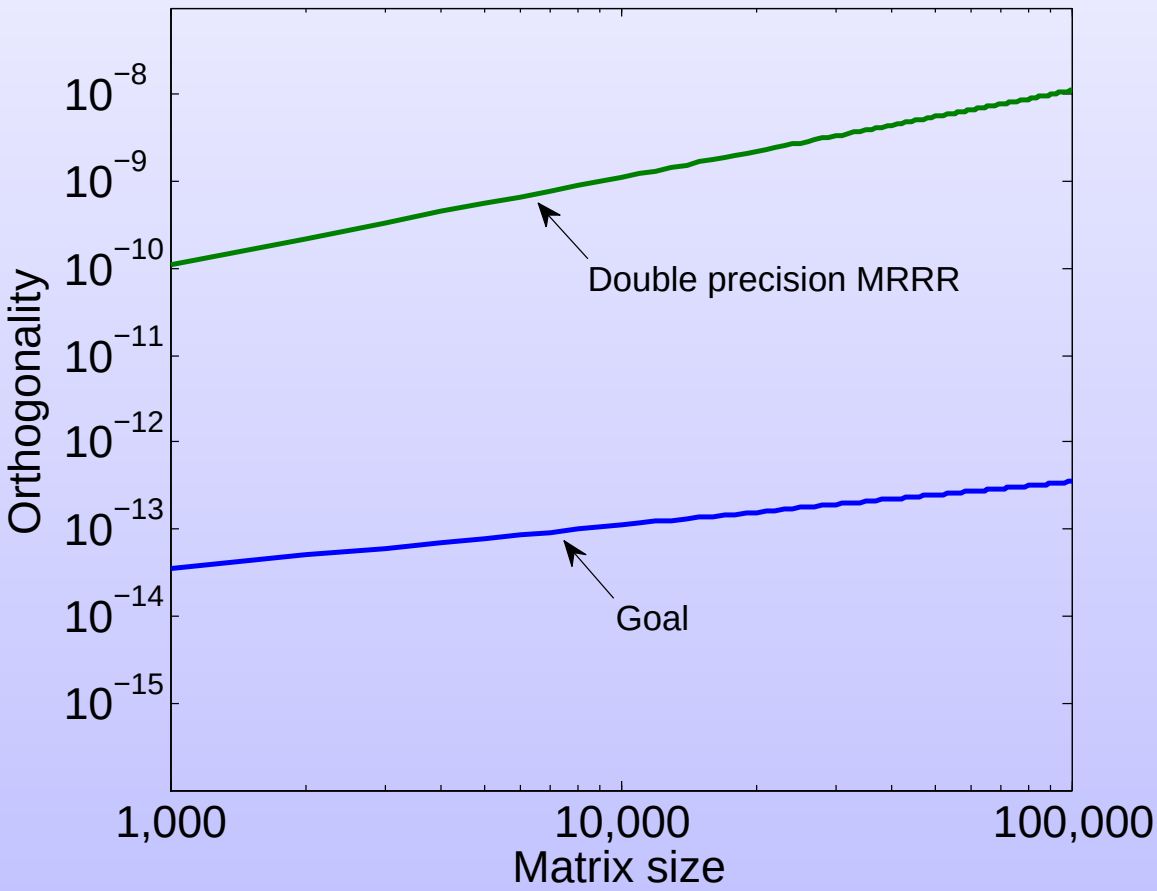
Paul Willems' dissertation $\Rightarrow$ improved robustness

But: $\begin{cases} \text{orthogonality(DC)} & \approx O(\sqrt{n}\,\epsilon) \\ \text{orthogonality(MR3)} & \approx O(1000\, n\, \epsilon) \end{cases}$

Basic idea

$$|\hat{z}_i^H \hat{z}_j| \leq C_1 (k_{rs} + k_{rr}\, d_{max}) \frac{n\epsilon_d}{gaptol} \leq C_2 \frac{n\epsilon_d}{gaptol}$$

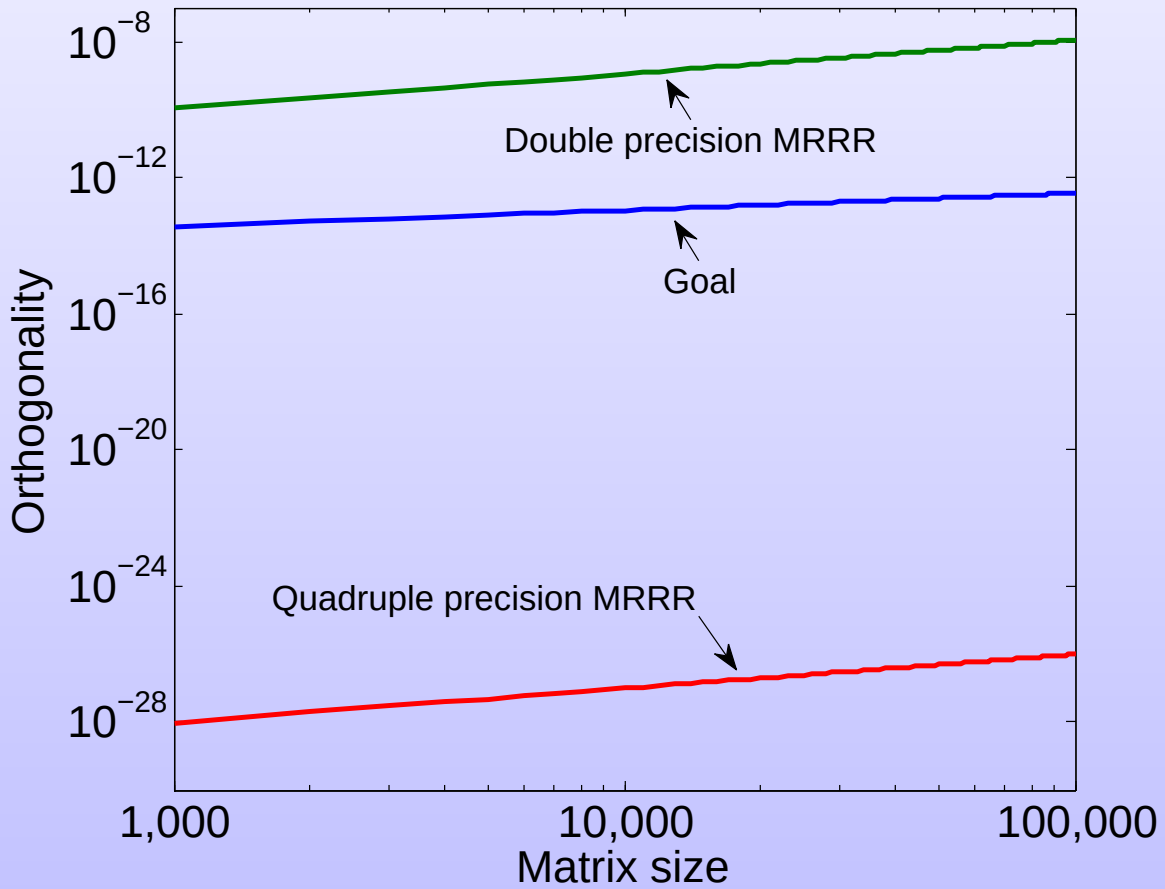
$$\epsilon_d \approx 10^{-16}$$



Basic idea: from ϵ_x to ϵ_y

$$|\hat{z}_i^H \hat{z}_j| \leq C_1 (k_{rs} + k_{rr}\, d_{max}) \frac{n\epsilon_q}{gaptol} \leq C_2 \frac{n\epsilon_d}{gaptol}$$

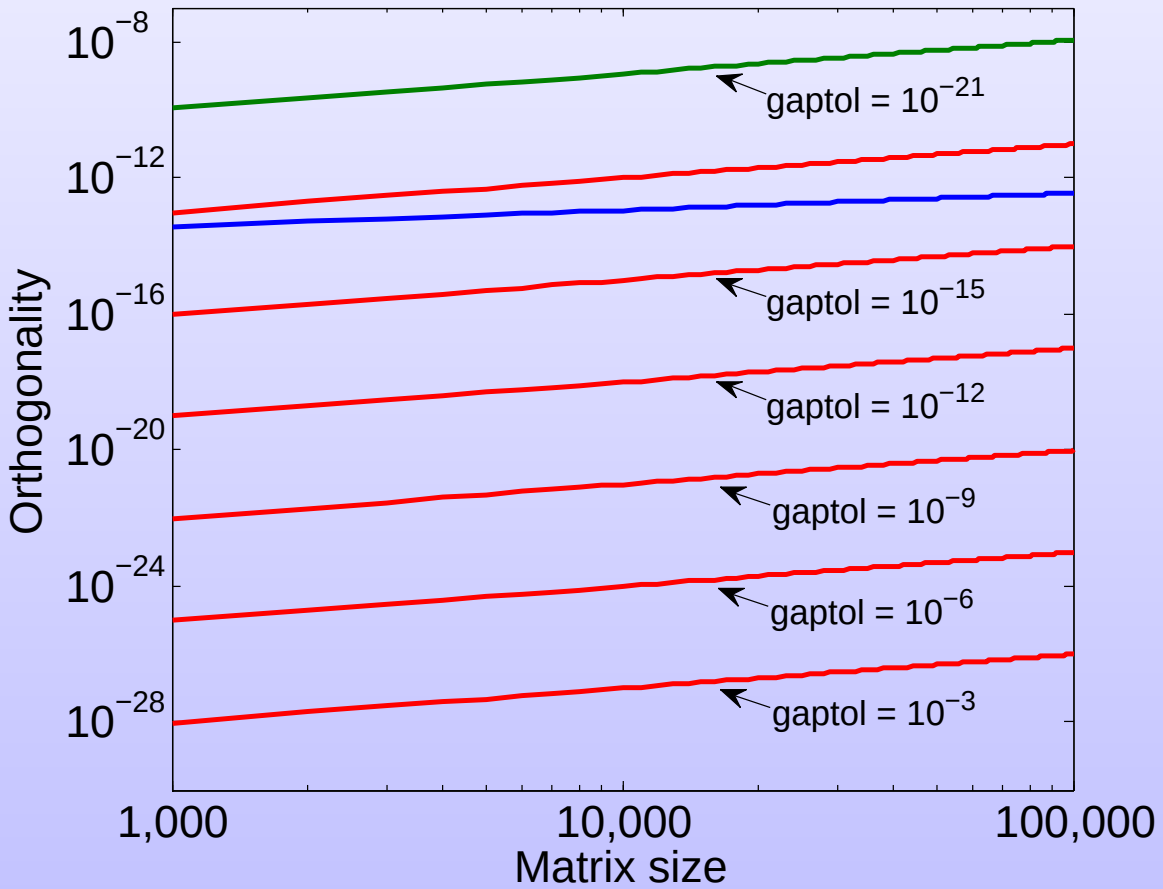
$$\epsilon_d \approx 10^{-16}, \quad \epsilon_q \approx 10^{-34}$$



Choice of gaptop

$$|\hat{z}_i^H \hat{z}_j| \leq C_1 (k_{rs} + k_{rr} d_{max}) \frac{n\epsilon_q}{gaptol} \leq goal$$

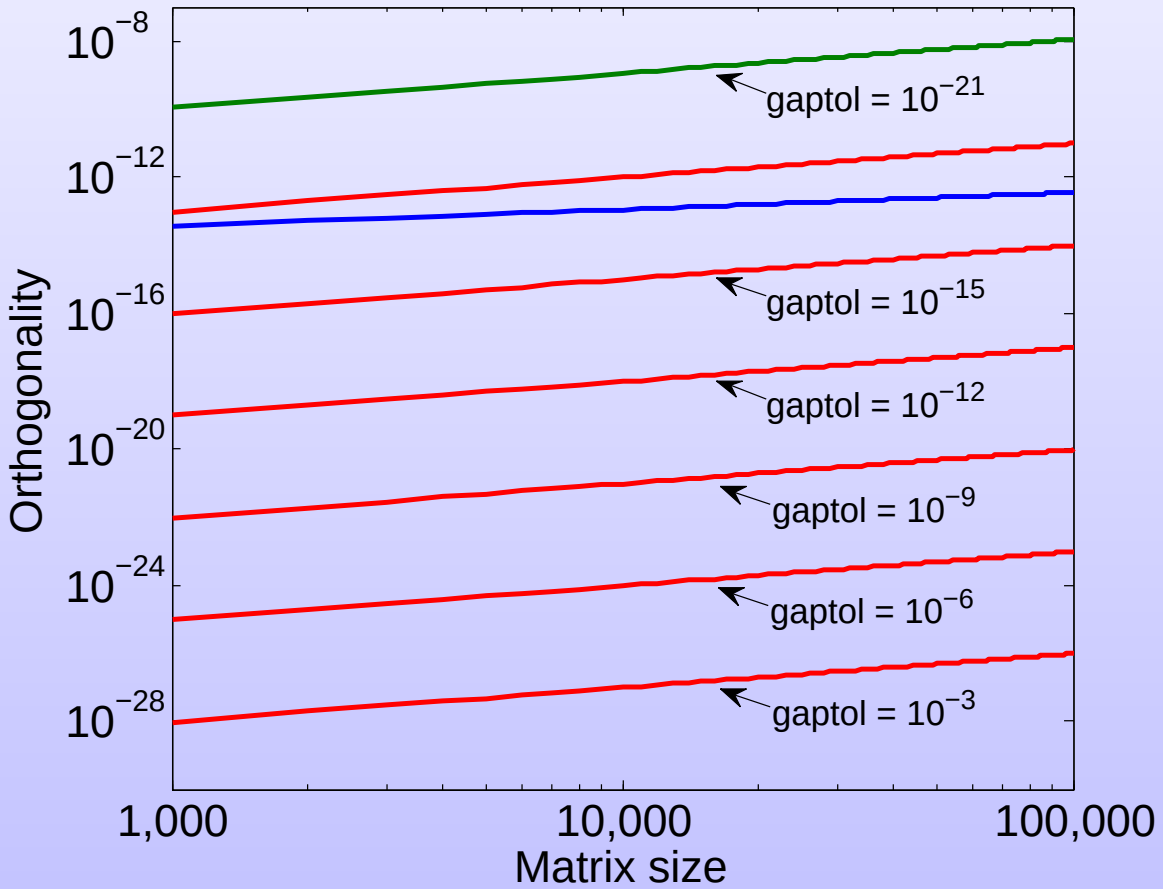
$$\epsilon_d \approx 10^{-16}, \quad \epsilon_q \approx 10^{-34}$$



Relax constraints: k_{rs} and k_{rr}

$$|\hat{z}_i^H \hat{z}_j| \leq C_1 \left(k_{rs} + k_{rr}\, d_{max}\right) \frac{n\epsilon_q}{10^{-10}} \leq goal$$

$$\epsilon_d \approx 10^{-16}, \quad \epsilon_q \approx 10^{-34}$$



Quad precision is expensive!

Goal: use high-precision only where it matters

$\Rightarrow$ Trade extra accuracy for speed, parallelism

Quad precision is expensive!

Goal: use high-precision only where it matters

$\Rightarrow$ Trade extra accuracy for speed, parallelism

ϵ_q to be used sparingly

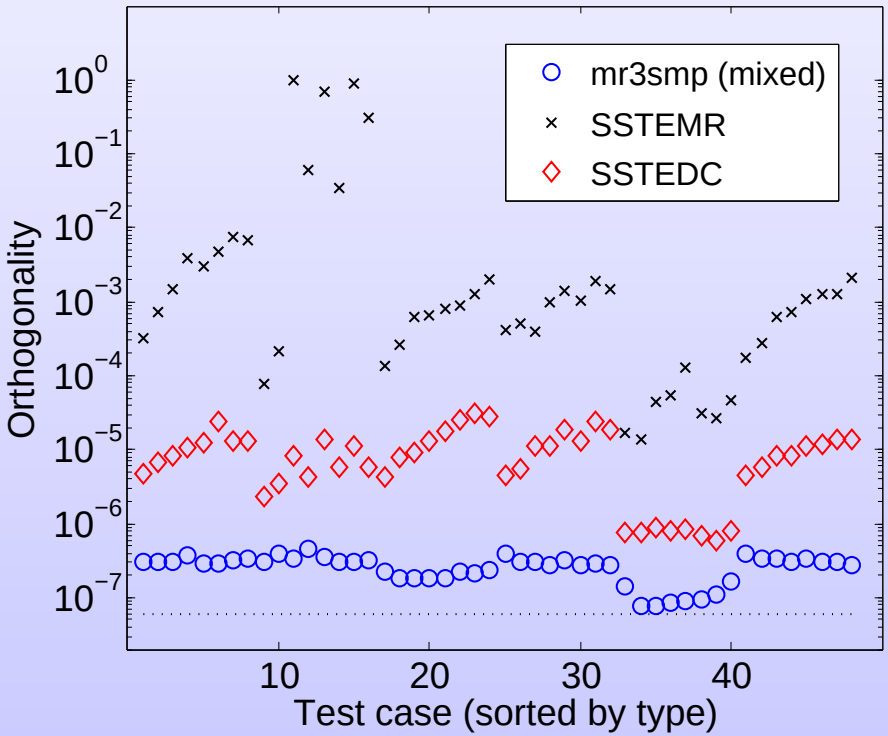
- initial eigenvalues: double
- refinement: double
- 1-eig (RQI): quad
- new RRR: quad

3 Results

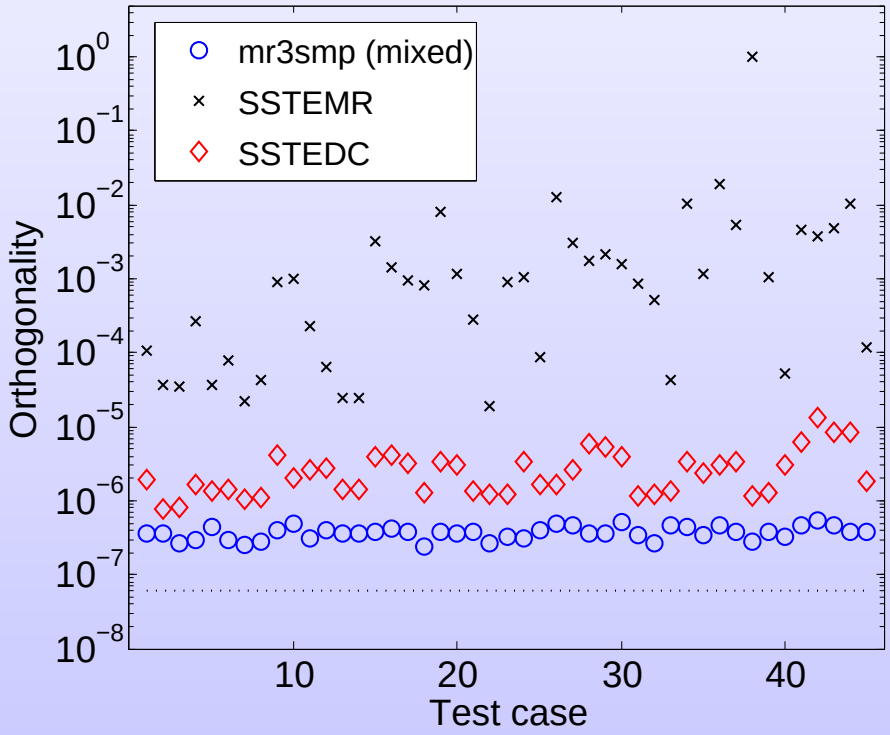
Tridiagonal – 1

single $\leftrightarrow$ double precision

artificial matrices
size $\in [2.500, \ldots, 20.000]$



application matrices
size $\in [1.000, \ldots, 8.000]$

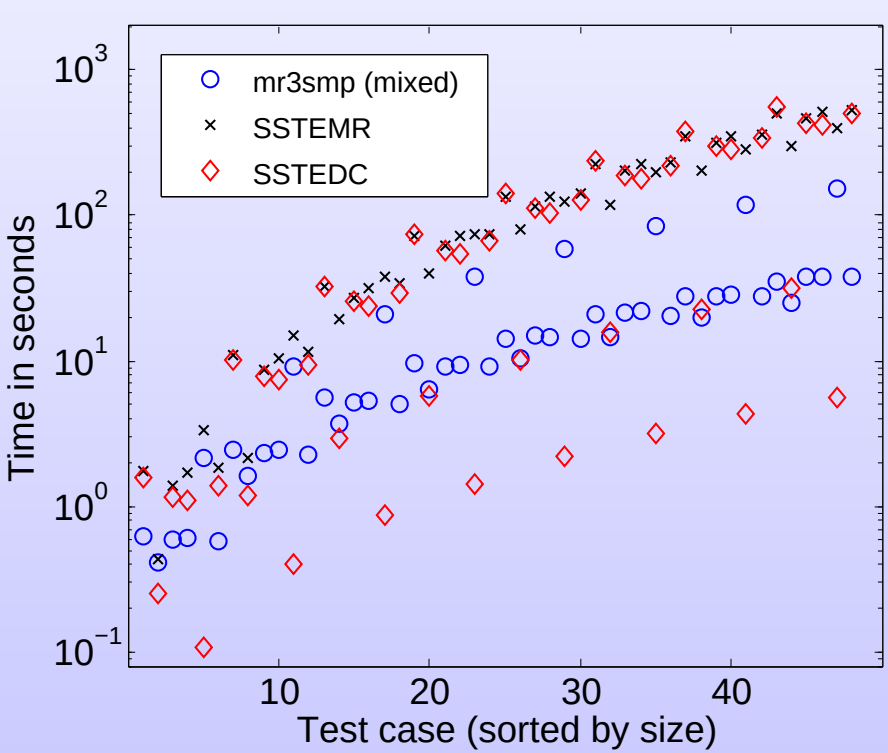


Orthogonality

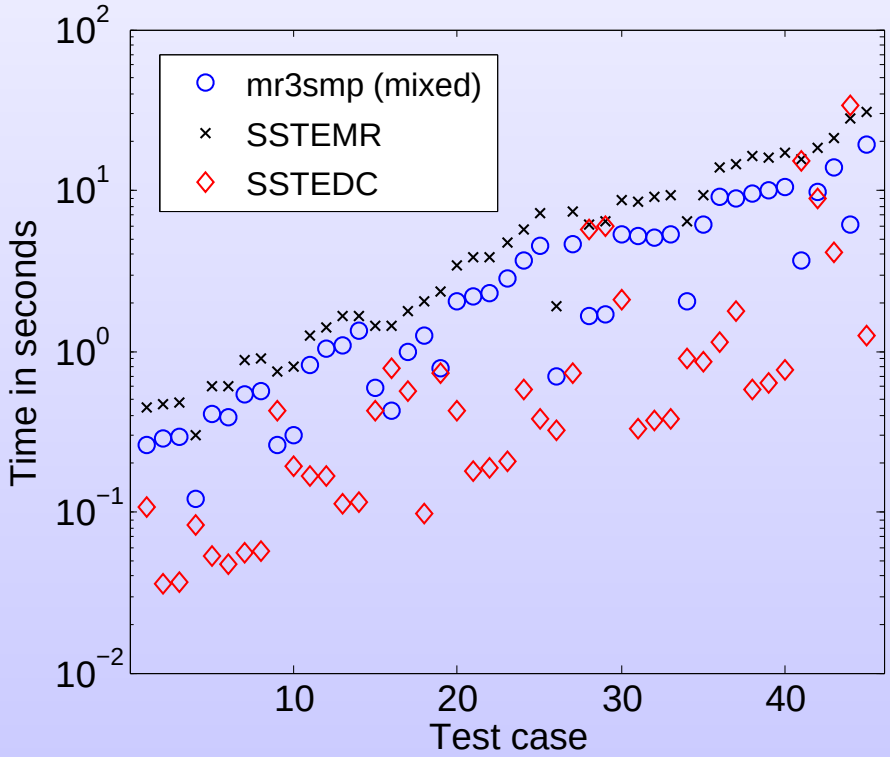
Tridiagonal – 1

single $\leftrightarrow$ double precision

artificial matrices
size $\in [2.500, \ldots, 20.000]$



application matrices
size $\in [1.000, \ldots, 8.000]$



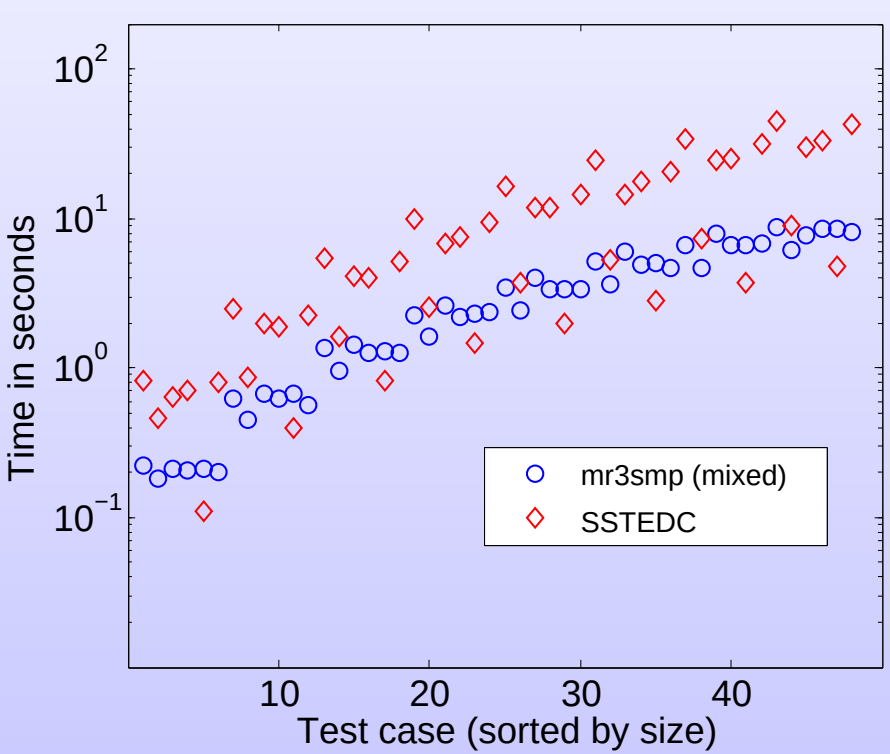
Execution time: 1 thread

Tridiagonal – 1

single $\leftrightarrow$ double precision

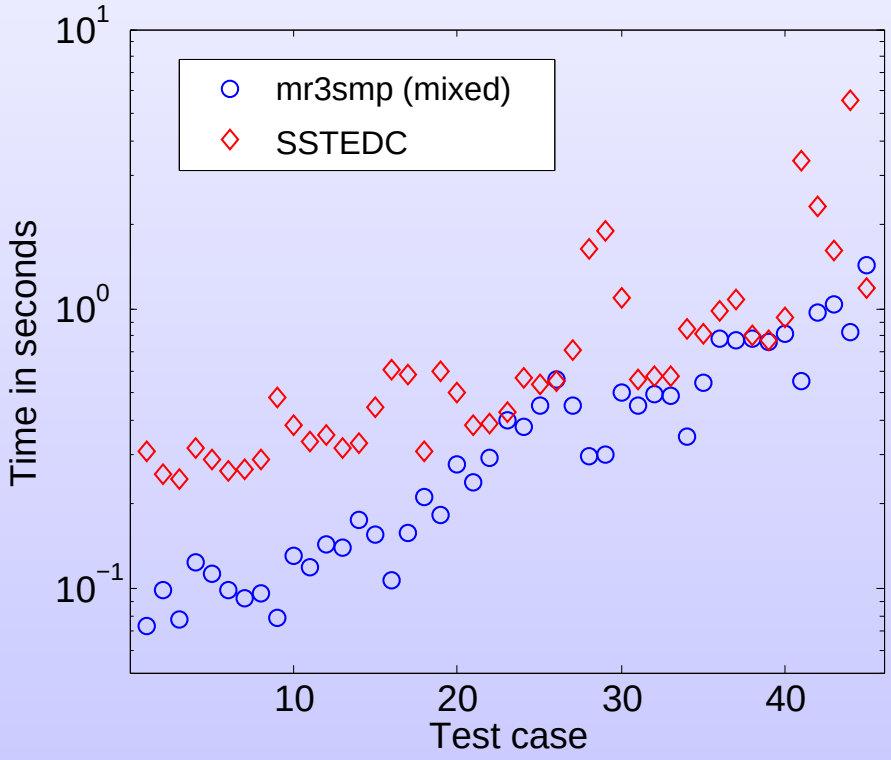
artificial matrices

size $\in [2.500, \ldots, 20.000]$



application matrices

size $\in [1.000, \ldots, 8.000]$



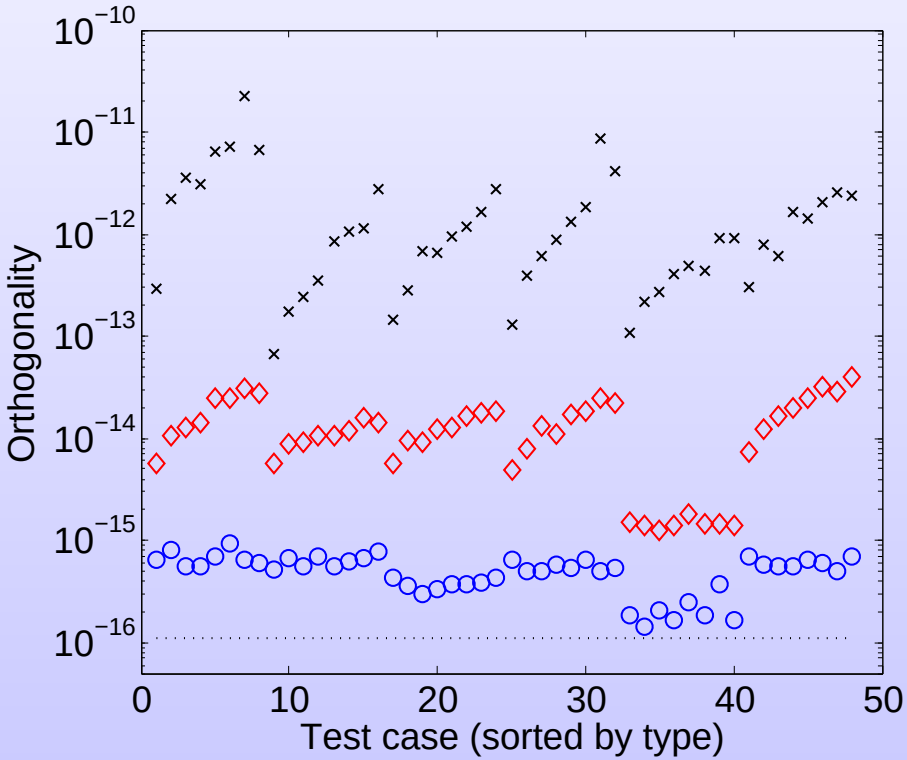
Execution time: 32 threads

Tridiagonal – 2

double $\leftrightarrow$ quad precision

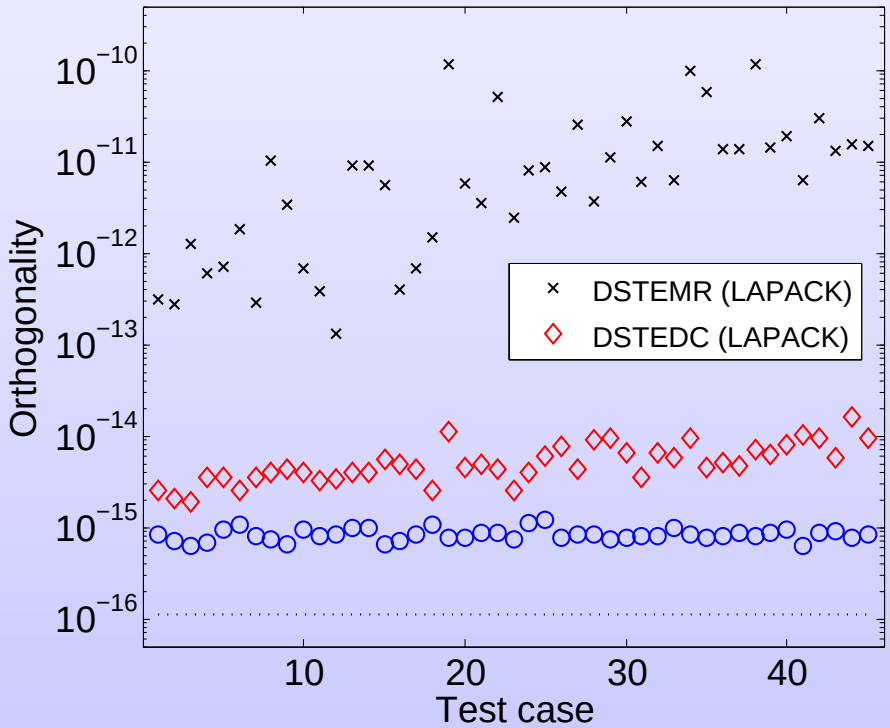
artificial matrices

size $\in [2.500, \ldots, 20.000]$



application matrices

size $\in [1.000, \ldots, 8.000]$



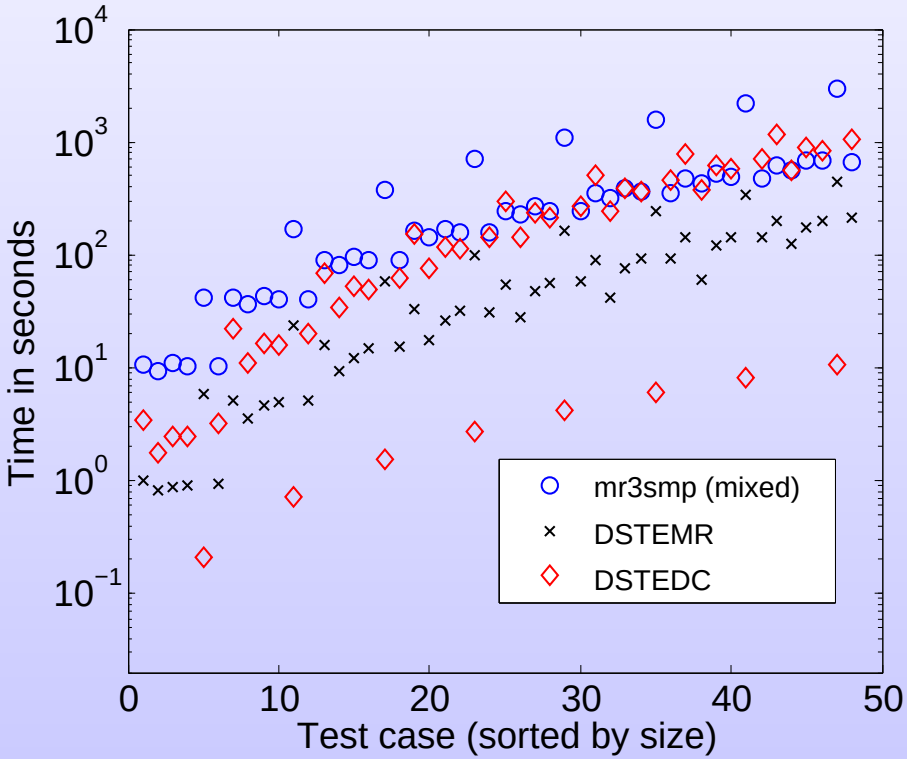
Orthogonality

Tridiagonal – 2

double ↔ quad precision

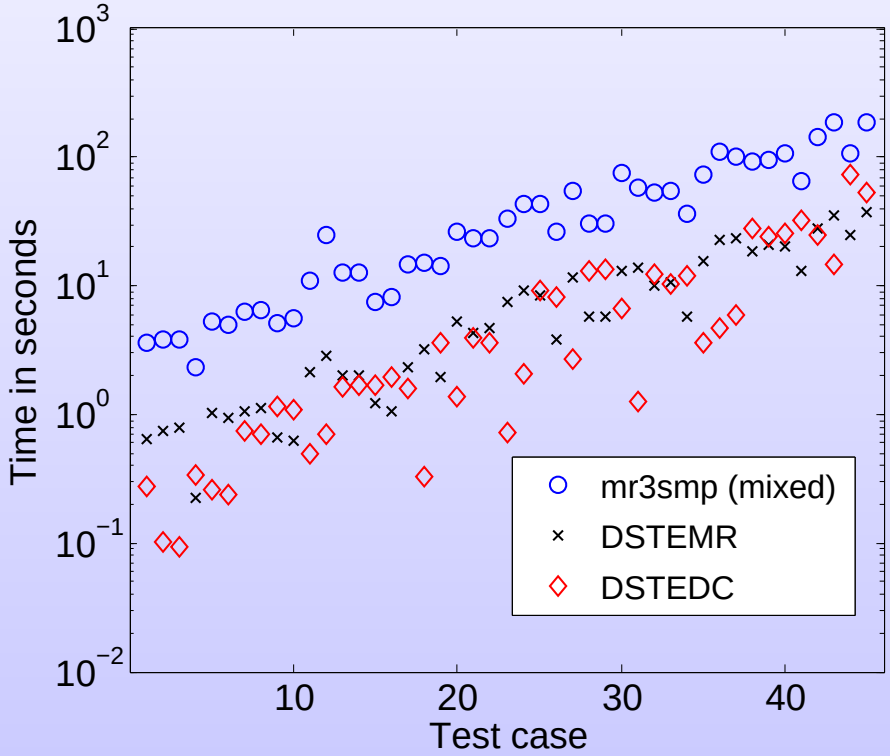
artificial matrices

size $\in [2.500, \ldots, 20.000]$



application matrices

size $\in [1.000, \ldots, 8.000]$



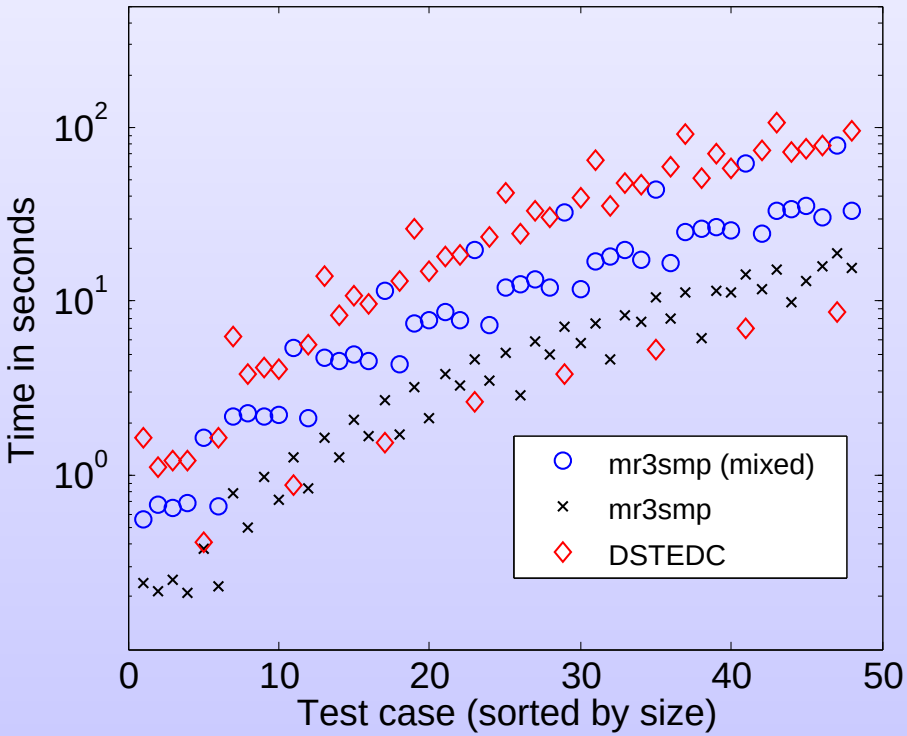
Execution time: 1 thread

Tridiagonal – 2

double $\leftrightarrow$ quad precision

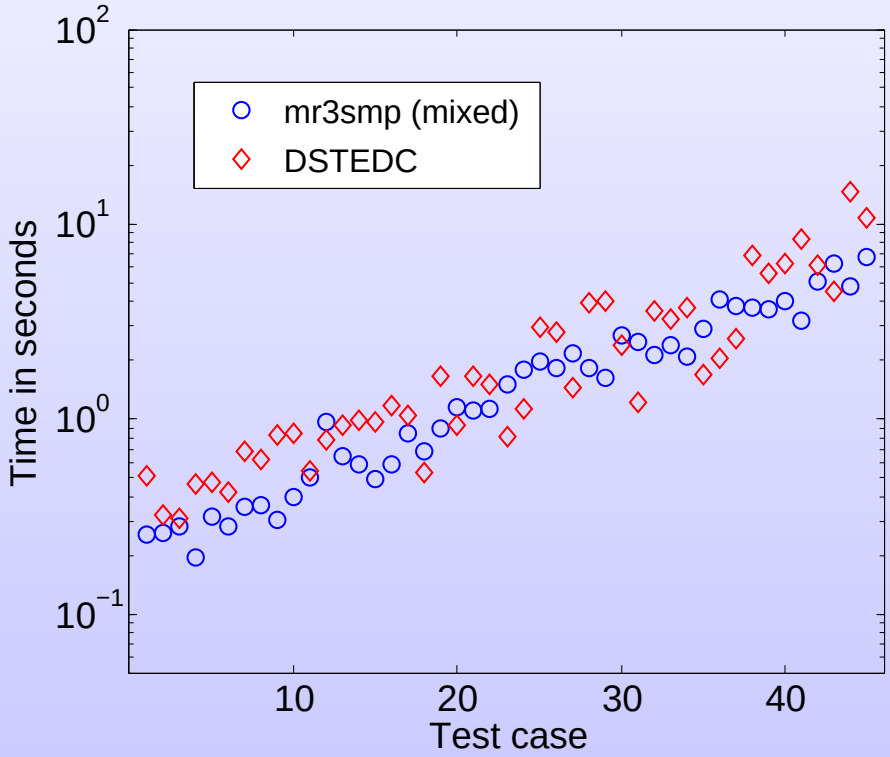
artificial matrices

size $\in [2.500, \ldots, 20.000]$



application matrices

size $\in [1.000, \ldots, 8.000]$



Execution time: 32 threads

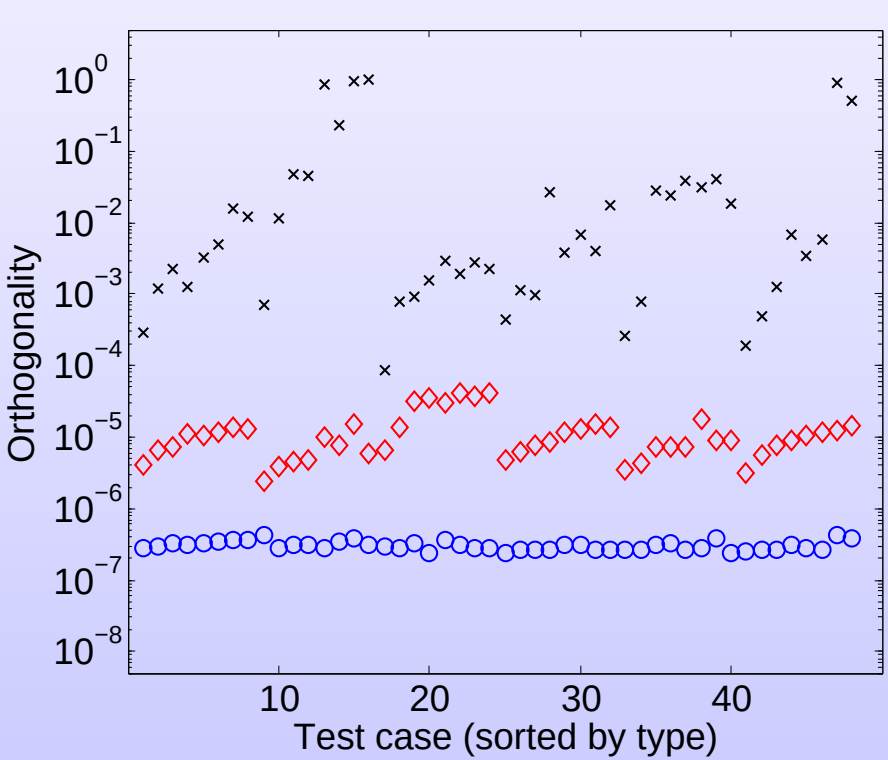
Results in context:
dense eigenproblems

Dense – 1

single $\leftrightarrow$ double precision

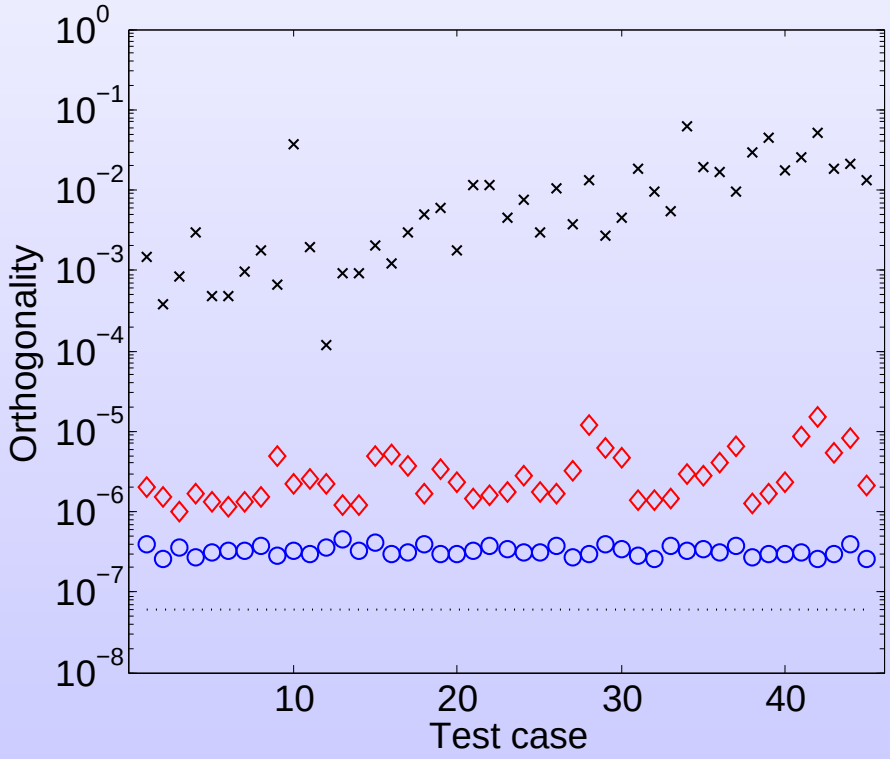
artificial matrices

size $\in [2.500, \ldots, 20.000]$



application matrices

size $\in [1.000, \ldots, 8.000]$



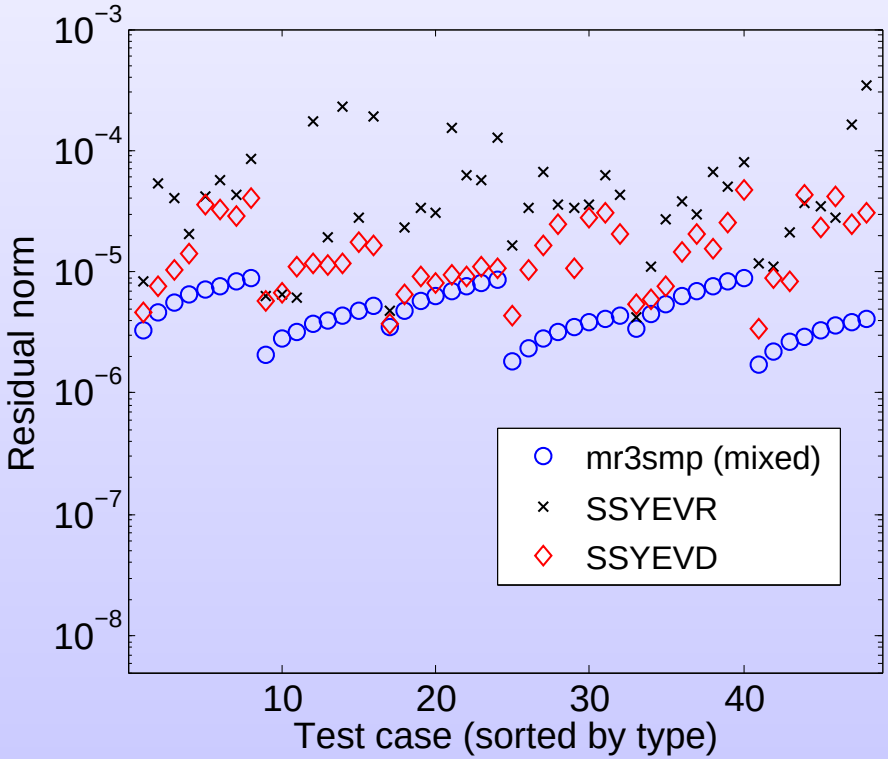
Orthogonality

Dense – 1

single $\leftrightarrow$ double precision

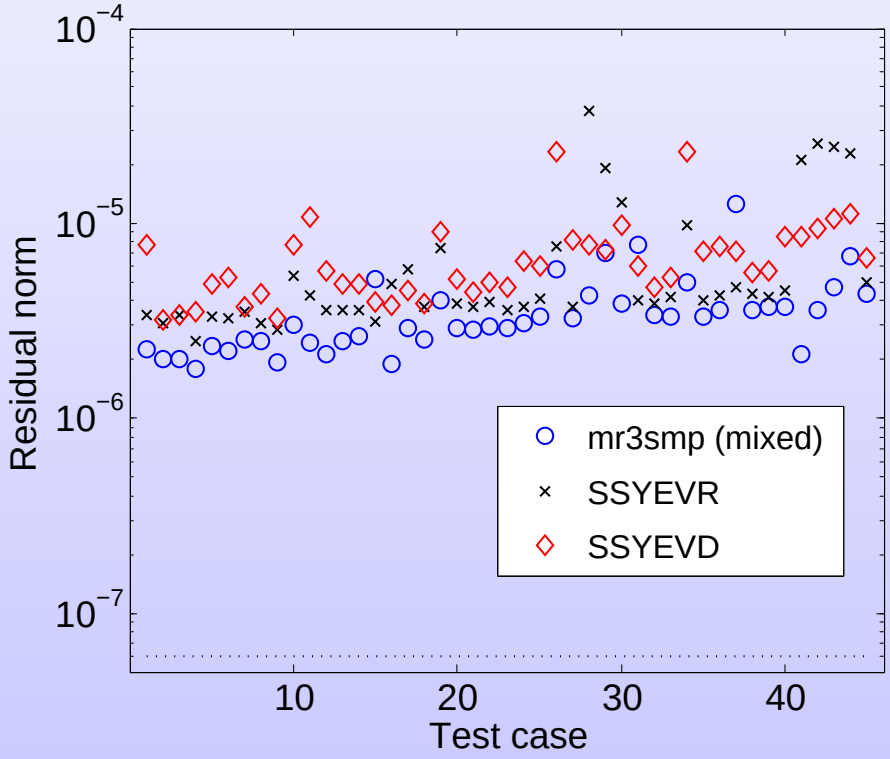
artificial matrices

size $\in [2.500, \ldots, 20.000]$



application matrices

size $\in [1.000, \ldots, 8.000]$



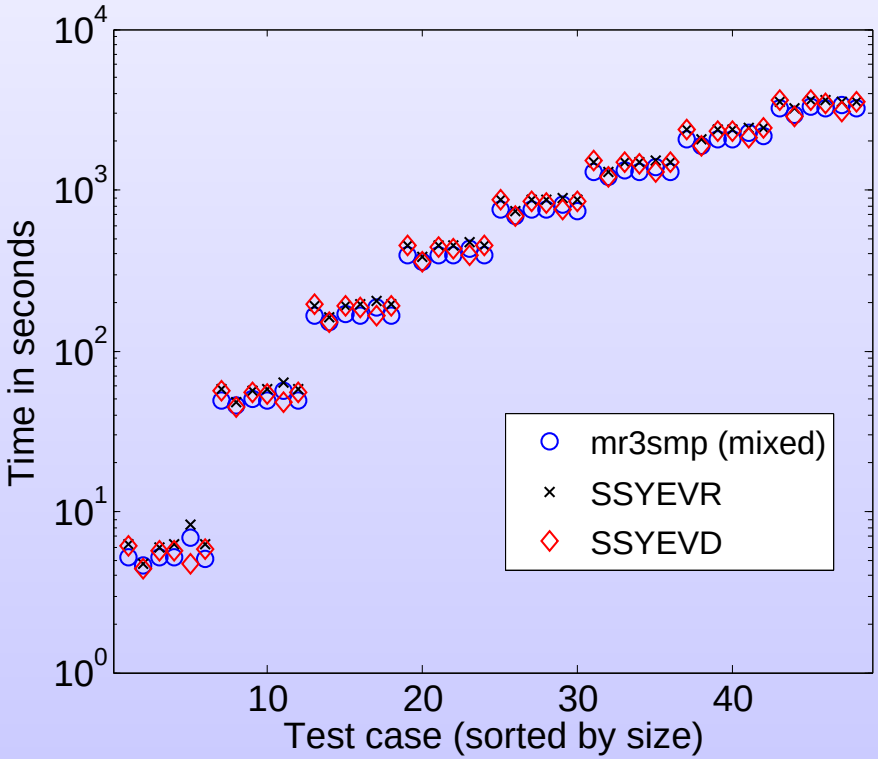
Residual

Dense – 1

single $\leftrightarrow$ double precision

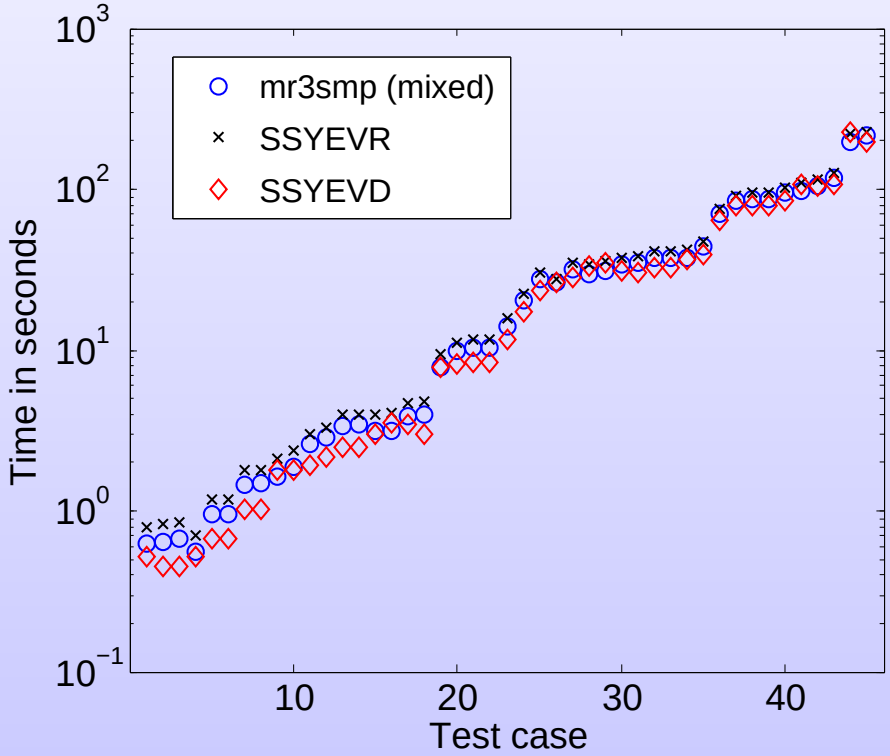
artificial matrices

size $\in [2.500, \ldots, 20.000]$



application matrices

size $\in [1.000, \ldots, 8.000]$



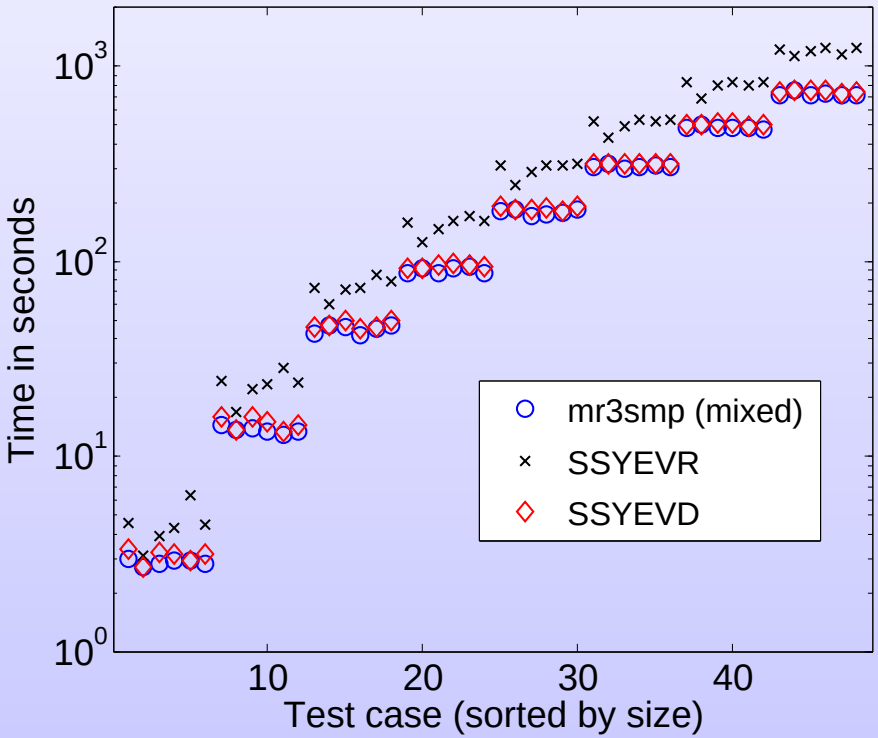
Execution time: 1 thread

Dense – 1

single ↔ double precision

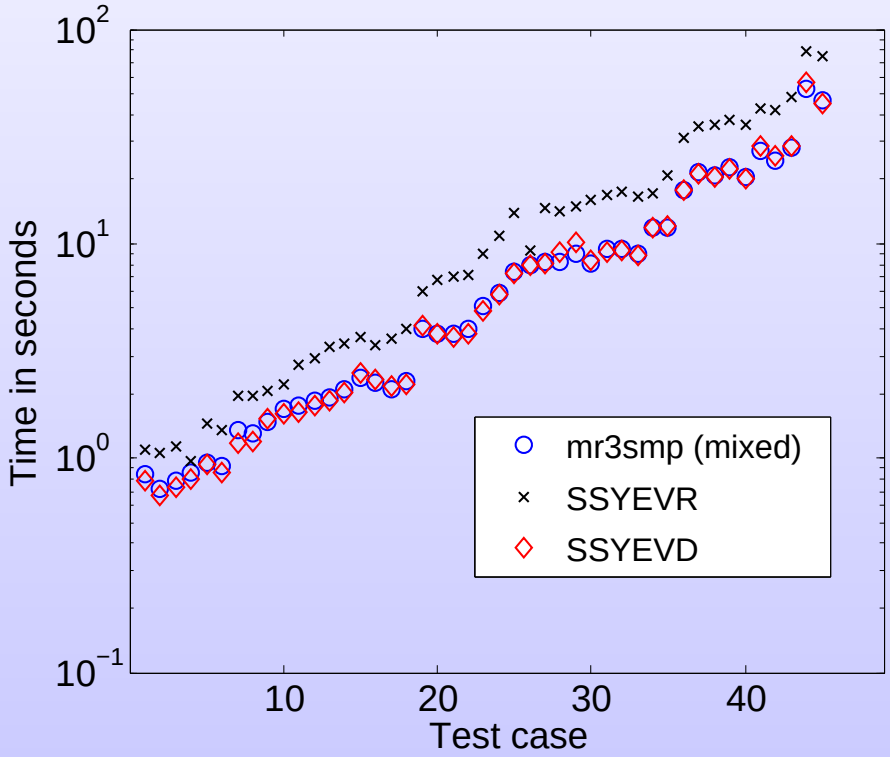
artificial matrices

size $\in [2.500, \ldots, 20.000]$



application matrices

size $\in [1.000, \ldots, 8.000]$

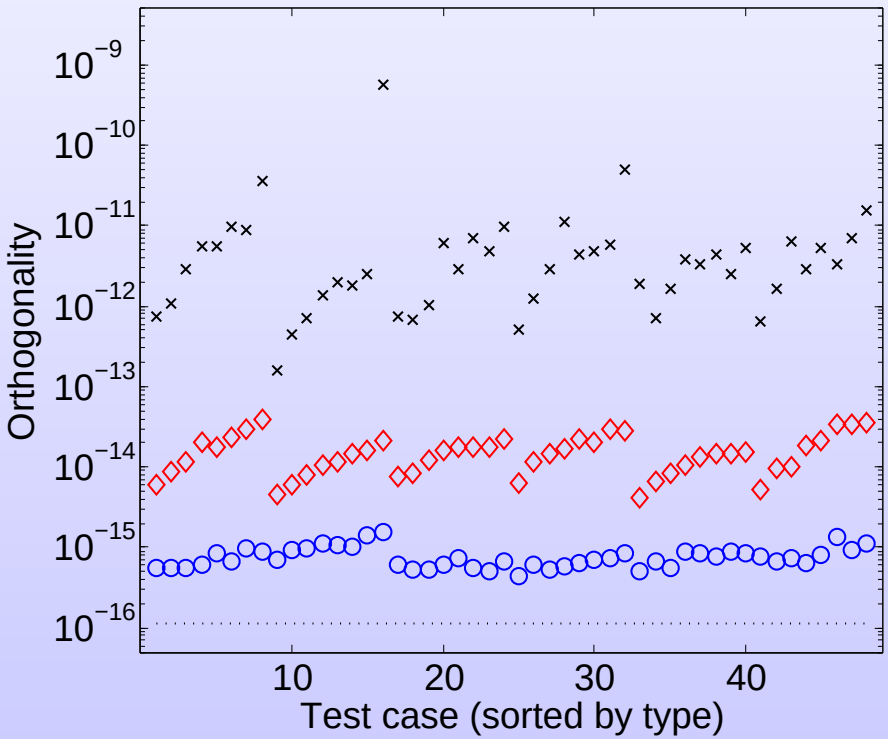


Execution time: 32 threads

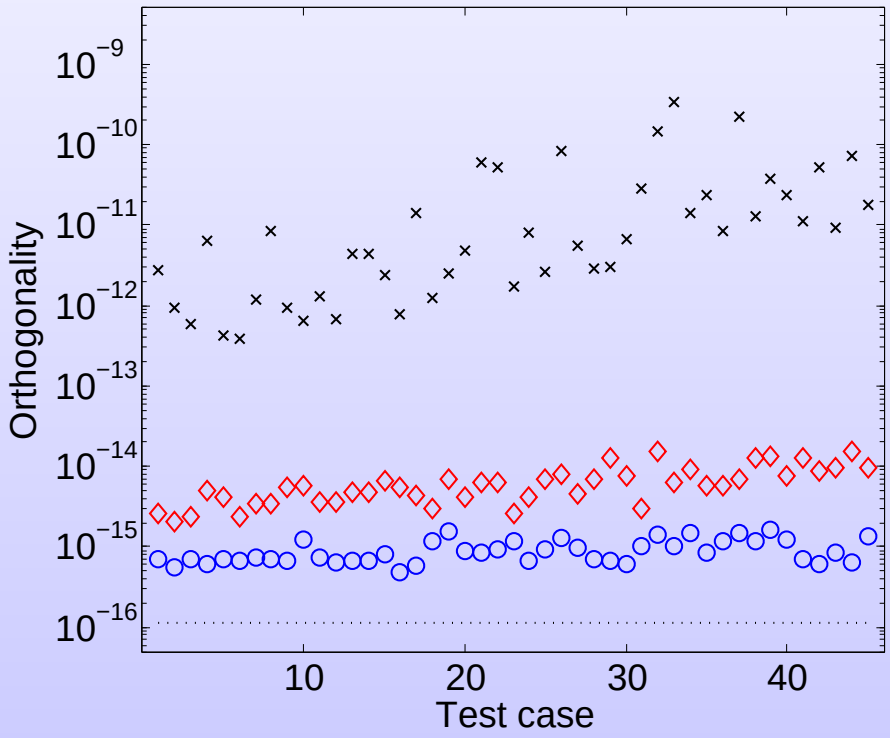
Dense – 2

double $\leftrightarrow$ quad precision

artificial matrices
size $\in [2.500, \ldots, 20.000]$



application matrices
size $\in [1.000, \ldots, 8.000]$

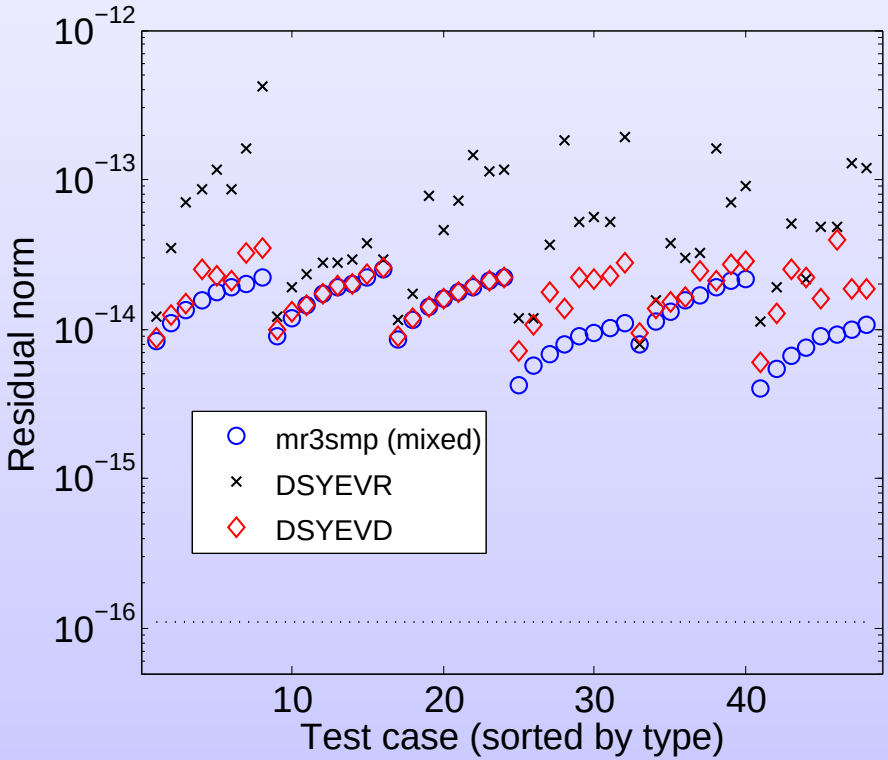


Orthogonality

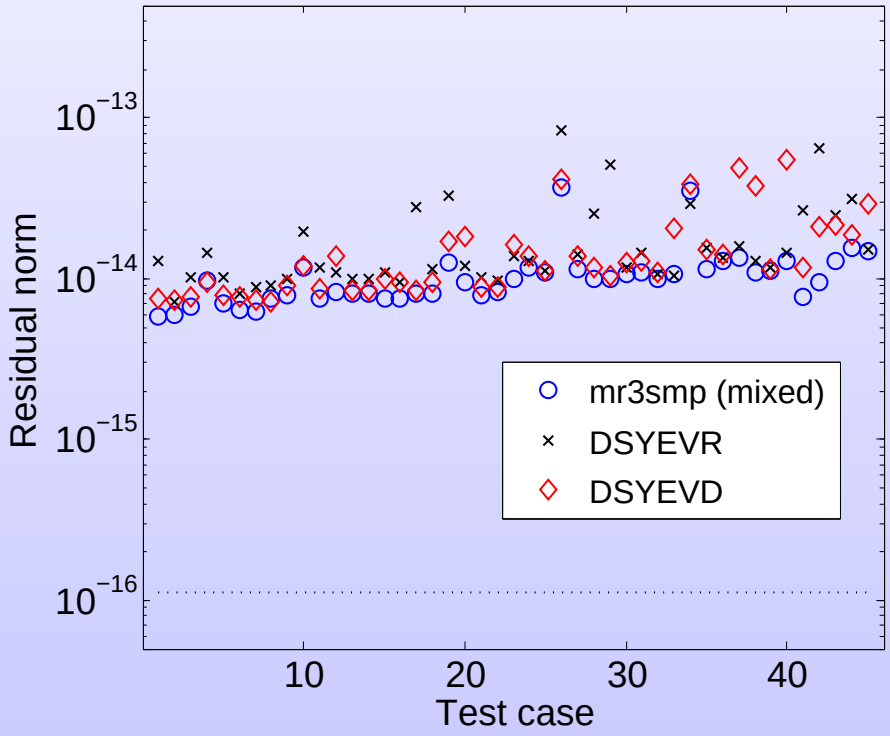
Dense – 2

double $\leftrightarrow$ quad precision

artificial matrices
size $\in [2.500, \ldots, 20.000]$



application matrices
size $\in [1.000, \ldots, 8.000]$



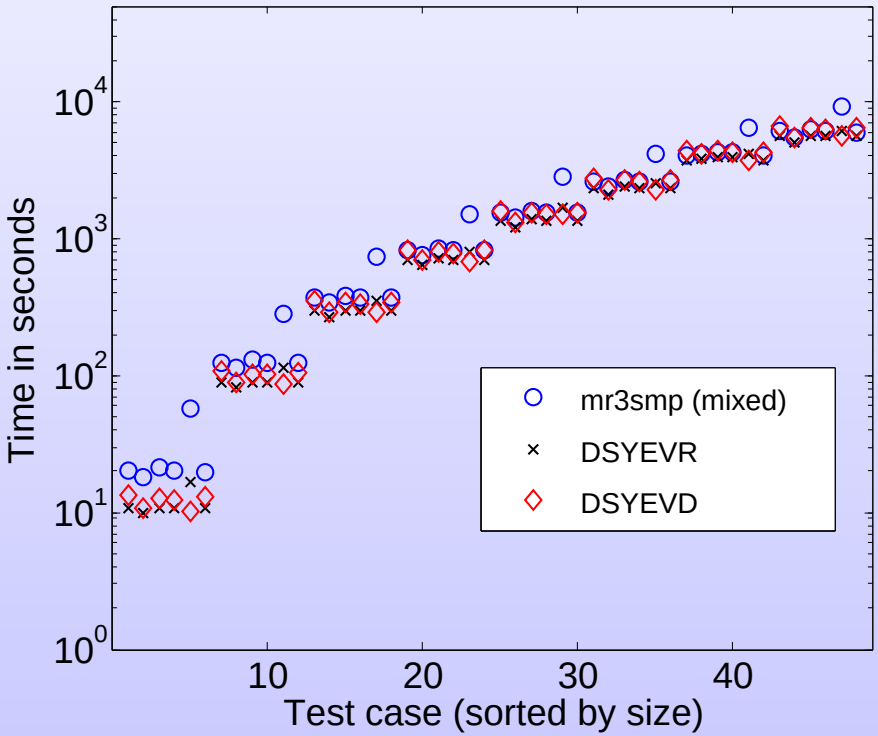
Residual

Dense – 2

double $\leftrightarrow$ quad precision

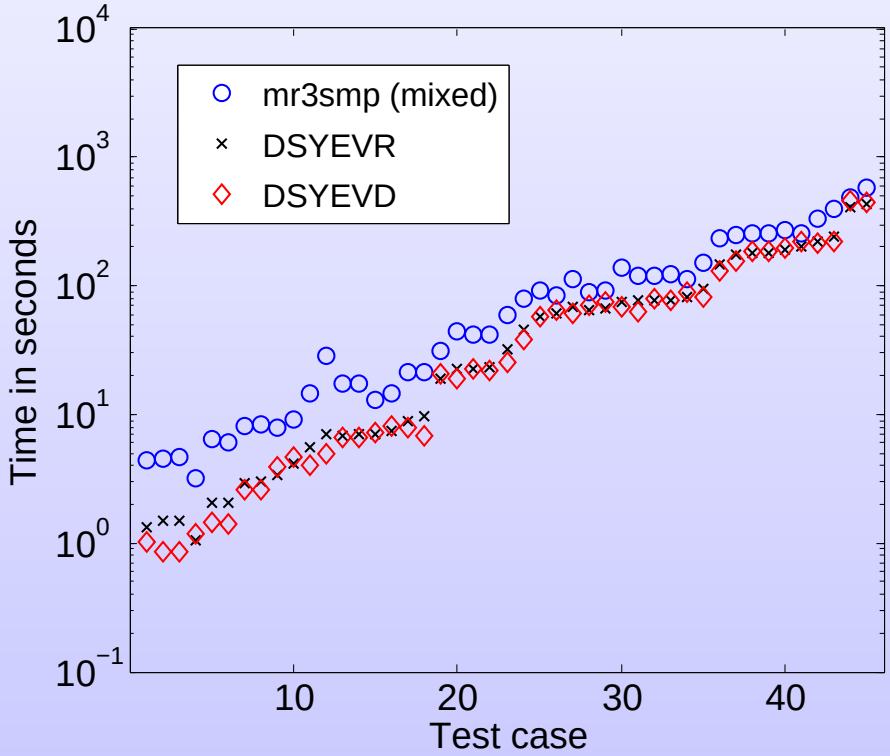
artificial matrices

size $\in [2.500, \ldots, 20.000]$



application matrices

size $\in [1.000, \ldots, 8.000]$

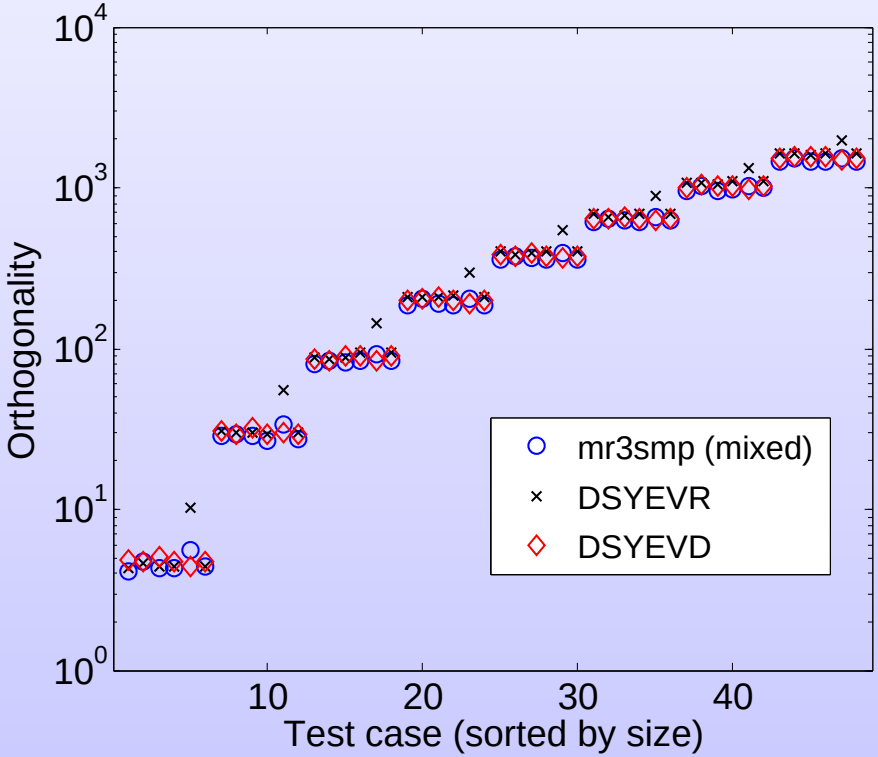


Execution time: 1 thread

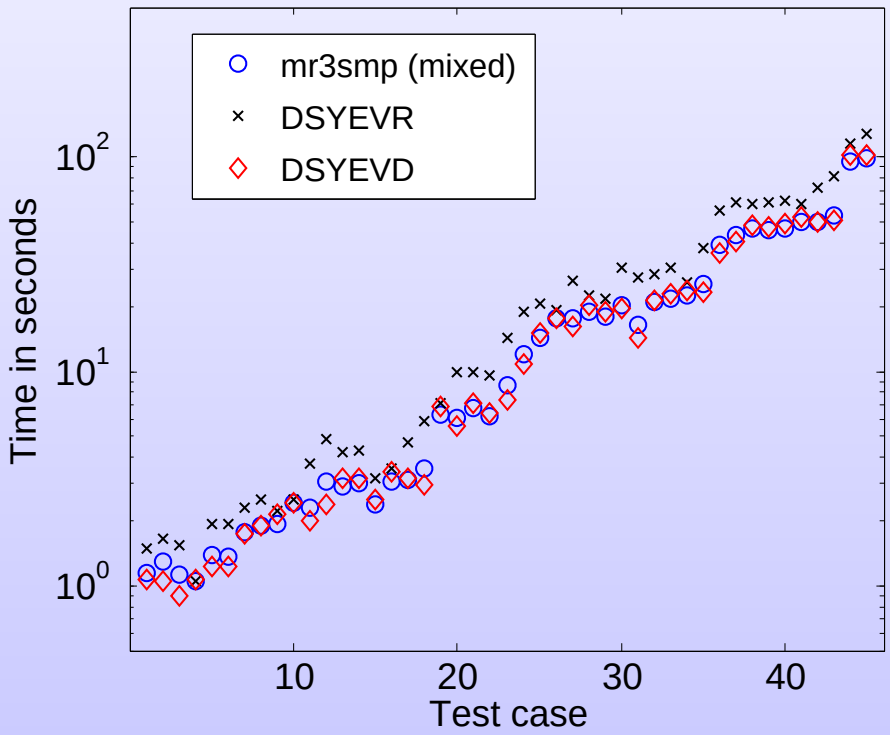
Dense – 2

double $\leftrightarrow$ quad precision

artificial matrices
size $\in [2.500, \ldots, 20.000]$



application matrices
size $\in [1.000, \ldots, 8.000]$



Execution time: 32 threads

Conclusions

Mixed-precision MR3

- MR3 becomes even more accurate than D&C and QR

Conclusions

Mixed-precision MR3

- MR3 becomes even more accurate than D&C and QR
- Increased robustness

Conclusions

Mixed-precision MR3

- MR3 becomes even more accurate than D&C and QR
- Increased robustness
- Generality
  well suited for large scale problems, subset computation, parallel executions, and dense and generalized problems

Conclusions

Mixed-precision MR3

- MR3 becomes even more accurate than D&C and QR
- Increased robustness
- Generality
  well suited for large scale problems, subset computation, parallel executions, and dense and generalized problems
- Even with software-simulated arithmetic,
  performance almost not affected
  - Parallelism greatly improved
  - Fewer operations performed
  - Single precision: more accurate and faster